FACULTY OF APPLIED SCIENCES
UNIVERSITY
OF WEST BOHEMIA

DEPARTMENT OF
COMPUTER SCIENCE
AND ENGINEERING

Bachelor's Thesis

# Visual Document Understanding

Marluce Quaresma

PILSEN, CZECH REPUBLIC

2024

FACULTY OF APPLIED SCIENCES
UNIVERSITY
OF WEST BOHEMIA

DEPARTMENT OF
COMPUTER SCIENCE
AND ENGINEERING

## Bachelor's Thesis

# Visual Document Understanding

Marluce Quaresma

**Thesis advisor**
Ing. Ladislav Lenc, Ph.D.

PILSEN, CZECH REPUBLIC 2024

**Citation in the bibliography/reference list:**
QUARESMA, Marluce. *Visual Document Understanding*. Pilsen, Czech Republic, 2024. Bachelor's Thesis. University of West Bohemia, Faculty of Applied Sciences, Department of Computer Science and Engineering. Thesis advisor Ing. Ladislav Lenc, Ph.D.

ZÁPADOČESKÁ UNIVERZITA V PLZNI
Fakulta aplikovaných věd
Akademický rok: 2023/2024

# ZADÁNÍ BAKALÁŘSKÉ PRÁCE

(projektu, uměleckého díla, uměleckého výkonu)

| | |
|---|---|
| Jméno a příjmení: | **Marluce WILL PIRES DOS SANTOS QUARESMA** |
| Osobní číslo: | **A20B0554P** |
| Studijní program: | **B3902 Inženýrská informatika** |
| Studijní obor: | **Informační systémy** |
| Téma práce: | **Porozumění vizuálním dokumentům** |
| Zadávající katedra: | **Katedra informatiky a výpočetní techniky** |

## Zásady pro vypracování

1. Seznamte se s úlohou porozumění vizuálním dokumentům (VDU – Visual Document Understanding).
2. Prostudujte dostupné metody a knihovny, které tyto metody implementují.
3. Seznamte se s dostupnými datovými sadami pro úlohu VDU.
4. Na základě provedené rešerše zvolte alespoň dvě metody a otestujte je na vybrané datové sadě.
5. Proveďte analýzu a srovnání výsledků použitých metod.
6. Zhodnoťte dosažené výsledky a navrhněte možná rozšíření.

| | |
|---|---|
| Rozsah bakalářské práce: | **doporuč. 30 s. původního textu** |
| Rozsah grafických prací: | **dle potřeby** |
| Forma zpracování bakalářské práce: | **tištěná/elektronická** |
| Jazyk zpracování: | **Angličtina** |

Seznam doporučené literatury:

Dodá vedoucí bakalářské práce.

| | |
|---|---|
| Vedoucí bakalářské práce: | **Ing. Ladislav Lenc, Ph.D.**<br>Nové technologie pro informační společnost |
| Datum zadání bakalářské práce: | **2. října 2023** |
| Termín odevzdání bakalářské práce: | **2. května 2024** |

L.S.

**Doc. Ing. Miloš Železný, Ph.D.**
děkan

**Doc. Ing. Přemysl Brada, MSc., Ph.D.**
vedoucí katedry

V Plzni dne 25. října 2023

# Declaration

I hereby declare that this Bachelor's Thesis is completely my own work and that I used only the cited sources, literature, and other resources. This thesis has not been used to obtain another or the same academic degree.

I acknowledge that my thesis is subject to the rights and obligations arising from Act No. 121/2000 Coll., the Copyright Act as amended, in particular the fact that the University of West Bohemia has the right to conclude a licence agreement for the use of this thesis as a school work pursuant to Section 60(1) of the Copyright Act.

V Plzni, on 1 May 2024

. . . . . .

Marluce Quaresma

# Abstract

Visual documents understanding has become important for many sectors such as business, education and healthcare since the images contain priceless information that needs to be extracted and processed. Due to the fact that the collected data plays a crucial role in decision-making, techniques for the digitization of information have been increasing. Moreover, document interpretation and further processing are two essential aspects that also contribute to the management of the previously mentioned sectors. Current Visual Document Understanding methods share a key feature called Optical Character Recognition (OCR), which involves converting images of text into machine-readable text. Lately, the interest in OCR-free methods has seen a rise in demand, because they directly analyse document images without the need for explicit character recognition. To address this topic, this thesis aims to analyse and compare OCR and OCR-free methods by training and testing on a specific dataset.

# Abstrakt

Porozumění obrazovým dokumentům je důležité pro mnoho odvětví, jako je obchod, vzdělávání nebo zdravotnictví, protože dokumenty obsahují neocenitelné informace, které je třeba extrahovat a zpracovat. Vzhledem k tomu, že získané detaily hrají zásadní roli při rozhodování, důležitost digitalizace informací a jejich dalšího zpracování se zvyšuje. Schopnost přesně interpretovat a zpracovávat vizuální dokumenty je take velmi důležitá. Současné metody pro porozumění obrazovým dokumentům jsou obvykle závislé na optickém rozpoznávání znaků (OCR), tedy převodu obrázků na strojově čitelný text. V poslední době je trendem používat tzv. end-to-end metody bez OCR. Tyto metody přímo analyzují dokumenty bez potřeby explicitního rozpoznávání znaků. Tato práce si klade za cíl analyzovat a porovnat metody vvyužívající OCR a metody bez OCR.

## Keywords

Visual Document Understanding • Optical Character Recognition • End-to-End Transformer

# Contents

# Introduction 1

Visual Document Understanding (VDU) refers to the competence of computers to understand visual details contained within scanned materials. They hold plenty of document elements and the VDU's main goal is to extract relevant information from them [1]. In other words, computers take images or graphs as inputs and return an editable file with the representation from the input. The research for such a method has increased significantly over the popularity of digital documents[1]. Despite the continued use of printed media in the business world, the physical space required for paper documents and the scanning of documents have encouraged the exploration of alternatives. VDU emerges as one of the most promising replacements for existing techniques. The process can be done in two approaches: Optical character recognition (OCR) and OCR-free.

Optical Character Recognition is a procedure that converts scanned images into machine-readable text output [2]. OCR is processed in 3 steps. The first one is the pre-processing, where the input file is divided into 2 parts - light areas, representing the background and the dark ones symbolizing the text. With the help of some other techniques that make the image perfect, the model performs text recognition and finally post-processing.

OCR-based methods are crucial members of practical applications in VDU [3] and have shown great performance in it [4]. However, it also brings some disadvantages, such as strictness in terms of language or types of documents and error propagation. OCR approaches not only carry high computational costs since they require three separate modules for text detection, text recognition and parsing [5], but also accumulate errors from both OCR and VDU interfaces For these reasons, the search for OCR-free mechanisms has increased.

Through the information's recognition, it is possible to execute various subtasks that facilitate a better comprehension of the document's composition.

- Document Classification

  Document types differ from each other in terms of visual style, format, organization, landmark, and spatial arrangement, among other features.[6] Given

the wide quantity of visual variabilities, the process of distinguishing the documents becomes challenging. Thus, document classification assigns a document to one of the predefined classes based on its content [7]. It tests whether the model can differentiate across different types of documents.

- Visual Question Answering (VQA)

  VQA is a type of deep learning that uses the combination of multiple forms of data to answer text-based questions about an image. Among the functionalities of VQA, document visual question answering is one of the most elaborate ones, since it requires the detection of objects and understanding of their relation so that an answer can be predicted.[8] Its main goal is to validate the further capacity of understanding of the mechanism.

- Key Information Extraction (KIE)

  Key information extraction is a useful tool in robotic process automation. On certain occasions, such as extracting the price of a product from a receipt, it is necessary that the model understands text in various layouts.[9] Consequently, KIE is responsible for extracting pre-defined key information[10] by using the composition of technical components from computer vision and natural language processing.

- Line Item Recognition (LIR)

  Line Item Recognition helps to resolve two problems of the VDU - text detection and text recognition.[3] In other words, algorithms for text detection have the goal of finding words or lines in the input files. Right after come the algorithms for text recognition, which aim to decode the textual information and extract the key information.[10] The type of line differs and therefore the algorithm can also be different. For example, P&ID uses continuous lines, lines incorporated with a line sign and a flow arrow.[11]

- Document Layout Analysis (DLA)

  DLA is the task that aims to identify the items within a document, analyse the way they are displayed and classify them into an appropriate category (e.g., graphs, text, figure).[12] Analysing the layout of a document can be very challenging since there are numerous aspects (e.g., writing style, document structures, and conditions) that differ from each document.[13]

- Named Entity Recognition (NER)

  Named Entity Recognition implies the identification and classification of entities written in texts, such as people's names, dates, and locations, among

others.[14][10] It belongs to the group of tasks needed to extract information and obtain its semantics from the text.

In the first part of this work, we start by presenting a list of OCR and OCR-free methods, followed by the dataset with which the models were trained and tested. Finally, there will be given details about the metrics that were used to evaluate the methods' achievements. The second part is focused on the training and testing of two models - one that incorporates OCR and another one which doesn't make use of the technology. The goal is to study systems that perform visual document understanding, analyse their implementations and afterwards their results. On account of that, three datasets will be chosen, with one of them being utilized in both models. Since it is a new approach researchers have been working on, the second mechanism will be tested with two extra datasets, so that its flexibility and adaptability to new scenarios can be tested. The results will be graphically studied and compared. Because of the difference between the datasets' structure and the structure expected by the model, the evaluation will be done not only based on the metrics, such as the Area Under the Receiver Operating Characteristic Curve, but also based on the comparison between the predicted text output and the truth ground. Lastly, the third part concerns analysing and comparing the output from both models using the dataset in common. The conclusion will contain suggestions for how the models and the dataset could be improved, to obtain better performances.

# Analysed methods 2

Specialists in the machine learning area came out with different methods for visual document understanding, using different techniques. In this chapter, some of the VDU methods are listed and their implementations are explained in detail.

## 2.1 OCR-free Document understanding transformer

Document understanding transformer (Donut) is an OCR-free method that consists of modelling a direct mapping from raw input to the desired output without OCR. In other words, it is an end-to-end technique based on a simple transformer-only architecture[5]. The first step of Donut is called pre-training. In this phase, the system learns how to read the input image by making predictions of the following words. The predictions are made by conditioning the image and the text contexts together.[5] Then, it learns how to understand the whole document according to the downstream task during the so-called fine-tuning phase. The entire process happens within the architecture, which is divided into two parts: a transformer-based visual encoder and a textual decoder. The process starts by dividing the image into quadrangular blocks called patches. The encoder extracts features from the given input image by converting it into a set of embedding values calculated from the number of image patches and the dimension of the encoder's latent vectors. Among the available models, Donut uses the Swin Transformer because it has the best performance. The encoder starts by splitting the image into patches, followed by applying the Swin transformer blocks to the patches. After that, the patch layers are merged and passed to the textual decoder. The decoder uses the architecture BART, which is responsible for generating a token sequence. Together, the transformers can convert the input images into editable files as figure 2.1 illustrates. One of the advantages of Donut is that it can support several languages. To prove that point, the dataset consists not only of real scanned documents in English but also in Chinese, Japanese and Korean, generated by the Synthetic Document Generator. Due to its representation capacity, JSON is the format adopted for the output.[5] To test the consistency of the

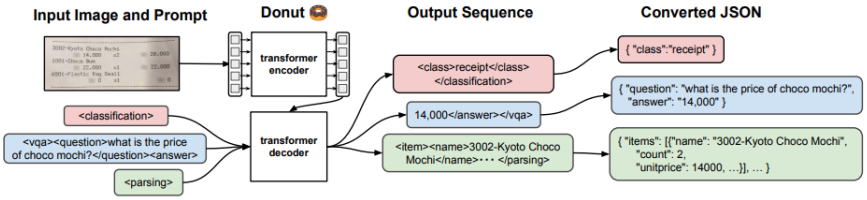


Figure 2.1: Conceptual Comparisons of Transformer Multi-Modal Encoder Architectures

system, 3 tasks are performed - document classification[7], document information extraction and document visual question answering[8].

It uses the RVL-CDIP dataset for the document classification subtask[7], which has a huge number of images with different classes. Based on the time and the accuracy observed, Donut was the fastest and most accurate mechanism[5], when compared with the OCR-based methods. The document information extraction is tested by the images from the Consolidated Receipt Dataset (CORD). Once again, studies have shown that Donut has the best performance. It has reached 84.1%[5] F1 score, which indicates highly evaluated Donut's precision and recall.

The document visual question answering requires a pair of document images and questions. After capturing both visual and textual information, it returns the answer to the question. The DocVAQ was the chosen dataset for this specific study. Although Donut has proven to be the fastest method, the LayoutLMv2 stood out in terms of similarities between the correct answers and the prediction results, reaching a score of 86.7%[5] of average normalized Levenshtein similarity.

## 2.2 Document Information Localization and Extraction Benchmark

Document Information Localization and Extraction Benchmark (DocILE) is an OCR method for document understanding which uses Key Information Location and Extraction (KILE) and Line Item Recognition (LIR).[10] The datasets used to study those 2 approaches are the UCSF Industry Documents Library and Public Inspection Files (PIF).

Business papers usually contain a list of numerous items. Some are represented as tables, where each column has specific details for an entity - products, people, among others. The selected dataset is divided into 3 subsets - an annotated set, an unlabelled set and a synthetic set. In other words, 2 groups of real business documents and another created by a generator with layouts similar to the first two. The dataset is selected according to some specifications such as the maximum number of

pages, the language and how old the paper is. From these papers and with the KILE and LIR methods it is possible to obtain a JSON file with the most important details. For the field annotation, there are some crucial aspects to take into consideration - location, field type, text and the line ID.[10] When processing documents with tables, the approach is slightly different, since the line item header contains the field types and the table grid contains the values for each field type. Besides that, there is also the metadata, which includes characteristics of the document type, page count, source, and page image sizes, among others. There are provided state-of-the-art transformer architectures, that cover text only (RoBERTa), image only (DETR) and multi-modal document representation (LayoutLMv3).

RoBERTa[15] is a model which uses different training schemes and tokenizers. LayoutLMv3 works with text, image and layout information jointly.[10] Images are firstly divided into non-overlapping patches and then the patches are fed to a linear projection layer. At the end, they are combined with positional embeddings. As for the texts, the tokens are combined with one and two-dimensional positional embeddings and after that, they are fed to the transformer model. DERT is an alternative method that performs object detection. The mentioned infrastructures use a joint multi-label Name Entity Recognition (NER)[14] formulation for both KILE and LIR tasks. The Line Item Recognition needs correct token classification of the classes and tokens assignment to individual Line Items. To this end, it is necessary to add classes, that represent the beginning <B-LI>, inside <I-LI>, outside <O-LI> and end <E-LI> of the line.[10] Apart from that, it is also important to reorder the OCR tokens in top-down and left-to-right order. The final predictions are structured by using some merging strategies. It starts by grouping the predicted tokens based on the membership to the predicted line item and follows by using the predicted OCR text lines to perform the horizontal merging assigned to the same class. Next, it draws a graph from the horizontally merged text blocks and finally merges the graph components. In other words, it joins the individual bounding text blocks and the text value.

The Line Item Recognition needs correct token classification of the classes and tokens assignment to individual Line Items. To this end, it is necessary to add classes, that represent the beginning <B-LI>, inside <I-LI>, outside <O-LI> and end <E-LI> of the line.[10] Apart from that, it is also important to reorder the OCR tokens in top-down and left-to-right order. The final predictions are structured by using some merging strategies. It starts by grouping the predicted tokens based on the membership to the predicted line item and follows by using the predicted OCR text lines to perform the horizontal merging assigned to the same class. Next, it draws a graph from the horizontally merged text blocks and finally merges the graph components. In other words, it joins the individual bounding text blocks and the text value.

The Average Precision (AP) shows that the content of the areas predicted by RoBERTa is more related to the document than those predicted by LayoutLMv3.

## 2.3 DocFormer

DocFormer is an end-to-end approach, which aims to execute 3 novel multi-modal tasks: learning-to-reconstruct, text describing image and multi-modal masked language. For that, it uses ResNet50, because it saves memory and connects text and visual features via spatial ones. When talking about the implemented concept, it is important to provide an overview of the architectures used in Transformer Encoder Multi-Model training. The 4 architectures are presented in the Figure 2.2.

The first architecture is the Joint Multi-Modal, which describes the fusion of vision and text into one long sequence - Figure 2.2(a). The disadvantage is the high costs for the cross-modality feature correlation. The second one is known as the Two-Stream Multi-Modal - Figure 2.2(b). As the name points out, it uses 2 branches one for each model. However, the features meet only at the end, which is not optimal. There is also the Single-stream Multi-Modal - Figure 2.2(c) - which mixes the features all together. Nonetheless, the combination is very risky, since vision and text components differ in the type of data. The last method is the Discrete Multi-Modal - Figure 2.2(d) - which joins visual, text and spatial features as residual connections to each transformer layer. This is also the one adopted by DocFormer.[4]

Before processing the visual, language and spatial elements, they are first passed through a preparation phase. For the visual items, lower-resolution visual embedding is extracted, based on a flattened sequence. The language components are prepared, by first treating text using a word-piece tokenizer and then feeding a trainable embedding layer. Spatial features' formation is done by getting the 2D bounding box coordinates from each word. Finally, all 3 variables are passed through a set of equations and the result is described as the sum of visual and language features is equal to the multi-modal one. The output is then provided to each layer of the transformer. Thenceforth, the realization of one of the pre-training techniques takes place.

The first one is the Multi-Modal Masked Language Modeling, which consists of predicting and reconstructing the entire sequence of text. The following pre-training is the Learn To Reconstruct, which differs from the first one by the simple fact that it produces image reconstructions. The last system is the Text Describe Image, which is instructed to identify whether a piece of text describes a document image. The usage of the mechanism described above is useful for sequence labelling, document classification, and entity extraction tasks.

When used with the FUNSD dataset, DocFormer can reach an 83.34% F1 score in the sequence labelling task. With the focus on entity–labelling, the method has

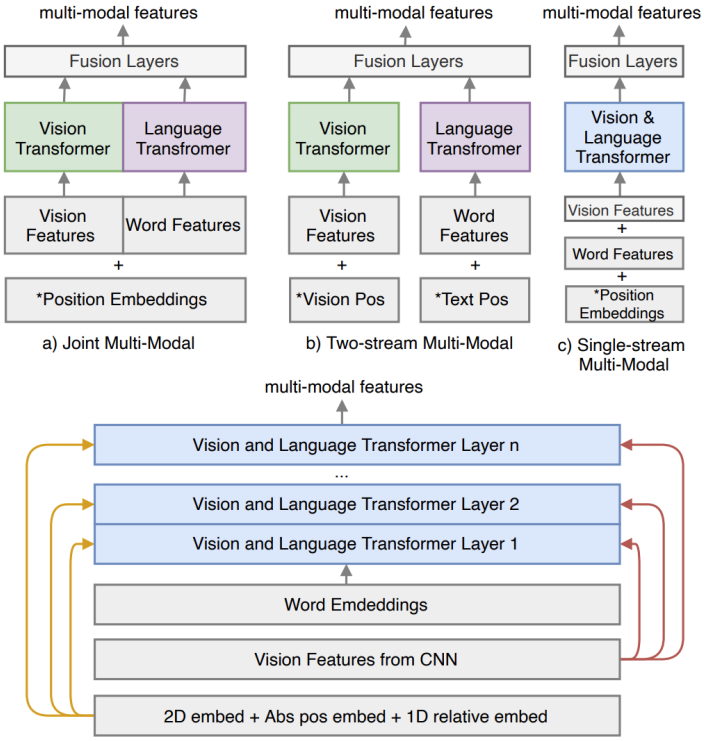


Figure 2.2: Conceptual Comparisons of Transformer Multi-Modal Encoder Architectures

shown to be the best and most effective among others - LayoutLMv2, BROS and LayoutLMv1. The same has been observed in the document classification task. Using the RVL-CDIP dataset has concluded that DocFormer performs better than the other models. A similar result was also seen in the entity extraction assignment. Given its complexity, it was tested twice with different datasets - CORD and Kleister-NDA datasets. The CORD dataset contains receipts and therefore the main purpose was the labelling of each word to the right field. The second dataset includes legal NDA documents. Thus, the task was to extract the values of 4 fixed labels. The F1 scores obtained were 96.99% and 85.8%, respectively.

## 2.4 Doc2Graph

Doc2Graph is a system, which is capable of using Graph Neural Networks (GNN) to study the hidden structure without any assumptions that the model can indeed describe it.

Thanks to its inherent representational power to describe elements and their pairwise relations, the information extracted from documents is organized in graphs.[16] The processing task starts by first using OCR and a pre-trained object detection

model to recognize words and detect the entities, respectively. The output objects of this step are represented by nodes. Then, a visibility graph [17] is used to build the relationship between the nodes – the edges. The strategy used seems to work well, but it contains some negative points. The process becomes complex because the generalization is not correctly done on different layouts and the edges' constitution strongly depends on the node detection process. To solve the issue, the system builds a complete graph and the task of identifying the crucial relations is set to the network. Although this methodology is the most efficient when it comes to problems such as Key Information Extractions [9], Document Layout Analysis [12] and Visual Question Answering [8], it is important to highlight the high computational cost regarding the memory and the training time that the large-scale document pre-training brings to the table. For that reason, Doc2Graph doesn't require a large amount of vision-language model pre-training models and focuses only on the recognition of the semantic text entities and their relationships. [16]

It is possible to obtain the nodes by applying methods with different modalities, such as visual, language and layout - 2.3: Input Project. The Doc2Grpach pipeline includes the spaCy large english model for the language model and the U-Net for the visual encoder. Regarding the edges, there were proposed 2 sets of features - a normalized Euclidean distance between 2 nodes and the relative positioning of nodes using polar coordinates.

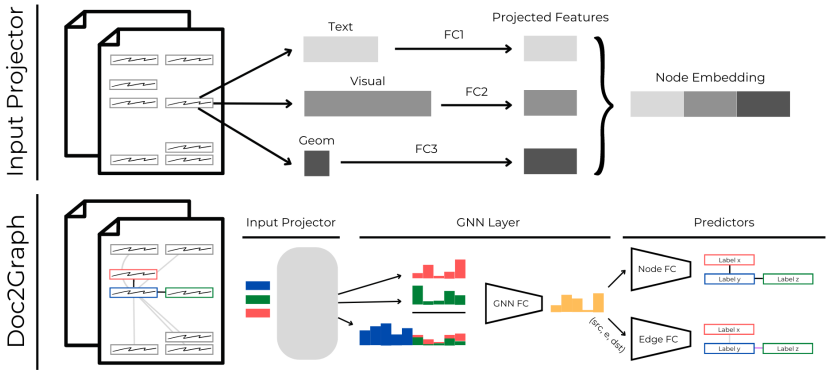


Figure 2.3: Doc2Graph framework

The final architecture makes use of 4 elements - figure 2.3: Doc2Graph. The first one is the Input Protector, which implements fully connected layers according to the modalities that are being used. The following component is the GNN [18] Layer. It redefines the aggregation method from a complete graph, by applying a set of equations. Then, there is the Node Predictor that draws the image of each node into the number of target classes. Finally, it uses the Edge Predictor to assign a label to each label.

To prove the entire design's capability, invoices from FUNSD and RVL-CDIP datasets are used. Tasks are performed for entity linking, layout analysis and table detections. The papers taken from the FUNSD dataset are characterized as grayscale images of numerous documents. The entities are detected using YOLOv5 and the graphs are created by connecting the ground truth. The results have shown that YOLOv5 performs better entity detection with an 87.2% F1 score, but the method Bert Relying On Spatiality (BROS) is better at entity linking. As for the RVL-CDIP invoices, studies have shown that Doc2Graph outperforms in layout analysis accuracy in terms of node classification and table detection.

# Datasets 3

## 3.1 CORD

The Consolidated Receipt Dataset (CORD) is a collection of Indonesian receipts that contains 8 superclasses - store, payment, menu, subtotal, total and others. There are also subclasses, which represent details of the superclasses. For instance, the store's subclasses include the name, address, telephone number and email.

The ground truth of one image has 3 attributes - meta, region of interest (ROI) and valid line. Meta is the overall information of the image - size, URL and id - Figure 3.1. The second attribute is a group of 4 coordinates (x,y) that encompass the area of the receipt - Figure 3.1: purple rectangle. Lastly, the valid line is a set of 4 coordinates and the content of each word - Figure 3.1: green rectangle. Because of its structure, the CORD dataset is used mainly for parsing tasks [19].

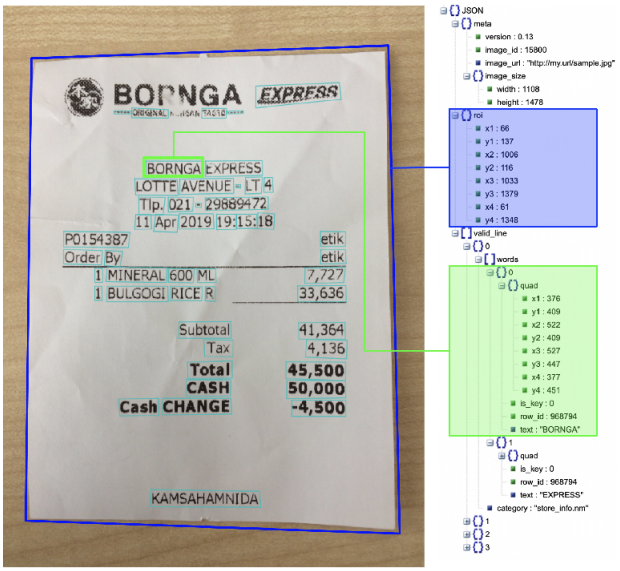


Figure 3.1: Image receipt from CORD and its extracted information

On account of the anonymisation of some sensitive information, like the cus-

tomer's name and the number of the card used for the payment, the documents have suffered some blurring - figure 3.2. Therefore, it is right to state that the documents are noisy [19], which makes the operation more difficult for models, such as DocFormer [4], DocILE[10] and Donut [5].


Figure 3.2: Example of noisy receipt from CORD

## 3.2 RVL-CDIP

Ryerson Vision Lab Complex Document Information Processing (RVL-CDIP) is a subset of the Illinois Institute of Technology Complex Document Information Processing Test Collection with over 400000 grayscale images. The files are defined as scanned of a big variety of documents [20]. There are in total 16 types, and among them, there are letters, emails and memos as demonstrated in figure 3.3. The files are distributed in 3 groups: 80% are for training purposes, 10% for validation and the remaining percentage for testing. Despite its low quality and resolution, the Ryerson Vision Lab Complex Document Information Processing (RVL-CDIP) dataset is used to evaluate the efficiency of frameworks, as DocFormer [4], Donut [5] and Doc2Graph [16] for classification tasks.

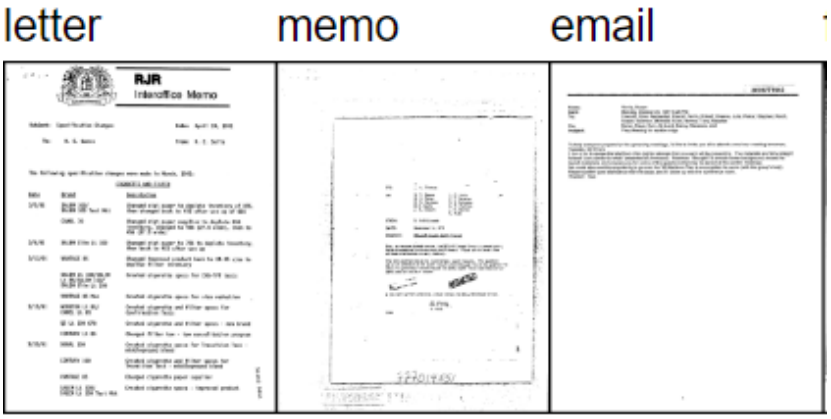


Figure 3.3: Examples of documents from RVL-CDI dataser

## 3.3 FUNSD

Form Understanding in Noisy Scanned Documents (FUNSD) is a small collection of scanned forms extracted from the RVL-CDIP dataset. The documents are noisy and differ in appearance [20], which makes them important since they help to test the visual document understanding method's capacity in different scenarios. The forms are encoded in JSON files, being represented as lists of semantic entities [20]. In one of the forms, the line "B.R. Pellett" represents the name of the Human Resources Personnel. The annotation shown in Listing 1 corresponds to the encoded JSON file containing information about the before-mentioned words. The text is described by an identifier, a label, a bounding box, a list of links with the id of other texts and a list of words [20].

```
{
    "text": "B. R. Pellett",
    "box": [257,169,357,185],
    "linking": [ [20,21] ],
    "label": "answer",
    "words": [ {"text": "B.","box": [257,172,272,185]},
    {"text": "R.","box": [272,172,290,185]},
    {"text": "Pellett","box": [300,169,357,183]}],
    "id": 21
}
```

Listing 1: Annotation of an element from a form of FUNSD dataset

DocFormer, DocILE and Doc2Graph applied the dataset to tasks such as text de-

tection, text recognition, spatial layout understanding, entity linking, and question-answer pair extraction [20].

## 3.4 SROIE

The scanned receipts OCR and key information extraction (SROIE) dataset is similar to the CORD dataset. It is a collection of scanned receipts, with the added detail that they are in English. The adopted structure is one of the reasons this dataset has been having a huge repercussion. It contains key text fields, ergo, their extraction makes the practices of archiving, indexing and document analytics easier. As a result, the dataset carries out indispensable tasks within the financial, accounting and taxation sectors.[21]

Figure 3.4 is an example of the PNG files from SROIE. Despite its physical quality, it is possible to extract some important from the document exhibited, for example, the products that were bought, their respective prices and the amount purchased.

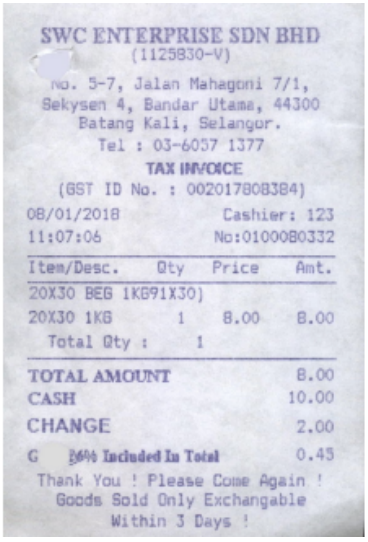


Figure 3.4: Examples of receipts from SROIE

## 3.5 Nomenclature dataset

Visual document understanding uses computer vision and natural language processing. Usually, tools that make use of such technologies are trained and tested in a way so that they can handle any type of document and still give the correct output for it. That being said, one of the most crucial aspects is the usage of datasets with which the models have never been trained.

In this experiment, there was used a set of map nomenclatures images along with their respective annotations. The images have a specific sequence of information, starting from the name of the area, followed by the cologne with both Roman and Arabic indication numbers and ending with the first and second sectors - figure 3.5. A particularity of this dataset is the fact that the characters' format in the documents is not as comprehensive as the one used in the previous dataset. Therefore, it poses challenges to the recognition and understanding of the information written. The way the details are ordered in the annotations can vary, but the labels are always the same - listing 2.

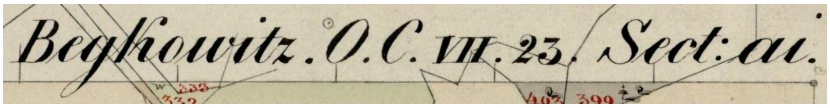


Figure 3.5: Examplef of nomenclature

```
{
    "area": "Begkowitz",
    "colonne": "OC",
    "roman": "VII",
    "arabic": "23",
    "first_sector": "a",
    "second_sector": "i"
}
```

Listing 2: Annotation from the nomenclature

Given the fact that Donut aims to understand not only computer-written characters but also handwritten ones, this dataset was designed with the purpose of testing such features.

## 3.6 Dataset comparison

To provide a comprehensive comparison of the previously mentioned datasets, the following table outlines their differences in terms of document type, number of files, JSON file structure, and preferred task.

| Dataset | CORD | RVL-CDIP | FUNSD | SROIE | Nomenclature dataset |
|---|---|---|---|---|---|
| Type of documents | Receipts | Various (letter, memo, email...) | Forms | Receipts | Map nomenclature |
| Number of files | > 1100 | > 400000 | 199 | 626 | 650 |
| JSON file structure | Meta, region of interest and valid line | Various | Semantic entities | Text fields | Text fields |
| Number of classes within the document | 8 | 16 | 4 | 4 | 6 |
| Task | Named entity recognition | Document classification | Text recognition | Key information extraction | Key information extraction |

Table 3.1: Datasets comparison

# Metrics 4

During the training and testing, tools pass through an evaluation phase. This step aims to tell whether or not progress has been made, by assigning a number to the results obtained. The evaluation part is important for both the programmer and user since it contributes to the improvement of the system and helps with the decision-making, respectively.

This chapter will feature four metrics - F1 score, Average Normalized Levenshtein Similarity, Average Precision and Area Under the Receiver Operating Characteristic Curve - along with their characteristics.

## 4.1 F1 Score

The F1 score is a machine-learning evaluation metric that rates a model's accuracy, by making a balance between two other classification metrics - precision and recall. The first evaluation measure symbolizes the percentage of positive predictions, which were indeed positive. The second one, also referred to as sensitivity, is the percentage of all relevant items that were correctly categorized as positive. At the end F1 score computes the number of times a model obtained a corrected prediction across the dataset, by calculating the harmonic mean.

$$F1Score = 2 * \frac{Precision * Recall}{Precision + Recall}$$

$$Precision = \frac{\text{True positive}}{\text{True positive} + \text{False positive}}$$

$$Recall = \frac{\text{True positive}}{\text{True positive} + \text{False negative}}$$

As it is shown in the formulas, from the 2 attributes previously identified the metric makes a robust gauge of model execution. It is one of the most used metrics since it provides solid results for balanced and imbalanced datasets.

This metric was used in all methods mentioned in the chapter 2.

## 4.2 Average Normalized Levenshtein Similarity

The average normalized Levenshtein similarity (ANLS) is an indicator, that targets a variety of tasks, such as information extraction and classification tasks. For instance, when performing the first task, some of the extracted information may contain small errors such as incorrect or misspelled words, and punctuation errors. Such errors should be subject to a different treatment when compared to completely wrong answers. Due to this fact, ANLS is a fundamental tool as it compares the similarities between the ground truth answers and the predicted results.[22] To quantify this similarity, the Levenshtein Distance is applied. The figure below is an example of its calculation.

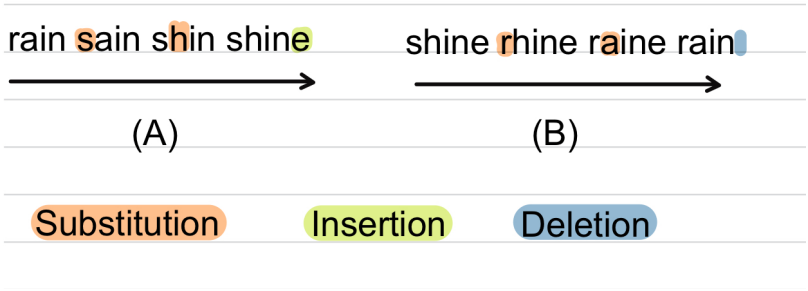


Figure 4.1: Examples showing Levenshtein Distance calculation

This distance is defined as the minimum number of operations - such as insertion, deletion and substitution - necessary to transform the predicted result into the ground truth. Assuming that the operations have the same cost of 1, figure 4.1 shows that the Levenshtein Distance between the words "rain" and "shine" is 3.

ANLS was the metric applied in Donut testing.

## 4.3 Average Precision

The Average Precision (AP) analyses whether the predicted area contains only the relevant characters. This parameter analyses the change of the values for precision and recall. It represents the area under the precision-recall curve.

As the graph in the figure 4.2 shows, average precision is high when both precision and recall are high, and low when either of them is low across a range of confidence threshold values, from 0 to 1.

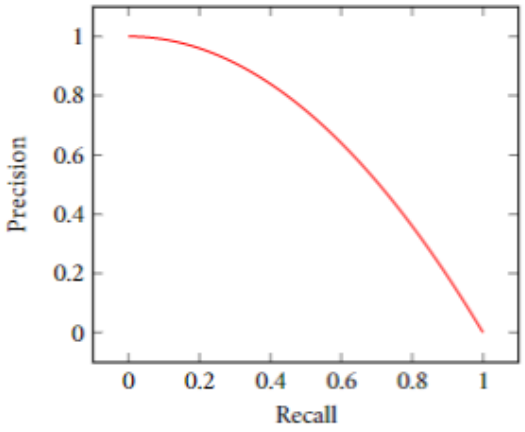


Figure 4.2: Average precision - variations of the values for precision and recall

## 4.4 Area Under the Receiver Operating Characteristic Curve

The Area Under the Receiver Operating Characteristic (AUC-ROC) curve is one of the most important metrics when it comes to classification problems since it provides a better way to summarize the information from the confusion matrices - figure 4.3 - obtained at different threshold settings.

The Receiver Operating Characteristic (ROC) is the curve that describes the relationship between the percentage of data correctly classified as positive - true positive rate or recall - and the percentage of data incorrectly classified as positive - false positive rate - for different thresholds. The formula for the second rate is shown below. Finally, the area under the curve of ROC (AUC) is the likeliness for the model to predict classes correctly.

$$\text{False Positive Rate} = \frac{\text{False Positives}}{\text{False Positives} + \text{True Negatives}}$$

| | | Actual | |
|---|---|---|---|
| | | Positive | Negative |
| Predicted | Positive | True Positives | False Negatives |
| | Negative | False Negatives | True Negatives |

Figure 4.3: Confusion matrix

This metric was applied to test the performance of Doc2Graph.

# Experiments 5

The research presented in this document aims to analyse the efficiency the visual document understanding methodology provides in accurately performing information extraction from images. For that, this chapter focuses on the training and testing of two models with distinct implementations. Therefore, the experiment structure will be divided into two sections, designed for each of the methods, and each section will be split into subsections according to the number of datasets the model will be tested on.

The first method to be analysed is Donut. As previously highlighted, researchers have developed an OCR-free framework to minimise the computational costs when performing VDU tasks. Thus, studying its performance helps to understand the impact of OCR on information digitization. The section related to Donut will be divided into three parts, representing the three datasets the model will use - SROIE, Nomenclature and FUNSD. During the development phase, one of the datasets utilized for testing this framework was the Consolidated Receipt Dataset [5]. Since one of the distinguishing features of CORD from SROIE is the language of the documents, trials will be done on the SROIE dataset to comprehend the effect of the language on the model's performance. Moreover, the Nomenclature set will be used to examine the model's performance on real-world data. Finally, FUNSD was chosen as one of the inputs to study Donut, so that its results could be compared to those from the trial using the OCR-based method.

The second method chosen to fulfil the goal of this study was Doc2Graph. Among the programs that incorporate OCR implementations, Doc2Graph has a particular organisational structure. Consequently, it was chosen for analysis to assess its effectiveness in extracting text details. The dataset chosen for evaluation is FUNSD, the same dataset used during the implementation phase [16].

Apart from the methods, some of the hyper-parameters were equally specified for both frameworks - source code 5.1.

Source code 5.1: Hyper-parameter configurations

```
self.training_args = Model(
            learning_rate=2e-5,
            per_device_train_batch_size=2)
```

Although the chosen value for the learning rate makes the model updates slower, it leads to higher stability and precision in the results. The second constant taken into consideration was the number of data samples in each training step. Due to memory limitations, it was necessary to select a small value, so that the required memory was smaller. Since this research has an inductive character, conclusions will be drawn from specific observations. So, the number of epochs for each training session will vary from zero to twenty to monitor the improvement of both models' performance and the way complex patterns are handled.

To analyse the models, the behaviour of two parameters - training loss and accuracy - will be studied across different values of epochs. The evaluation will be based on metrics, such as average Levenshtein distance and the AUC. In the end, the experiments performed for this document will be put side by side with the ones done by the developers of each method, so that the highlight points can be identified and described.

## 5.1 Donut

As stated in the section 2.1, Donut is a framework that does not require OCR modules, which proposes a less complex, but still skilful approach. This also means that when using the method, both the encoder and decoder transformers are trained and tested for the recognition of the characters and the specified task - figure 2.1 - respectively. For that reason, Donut was tried out along with 3 datasets - SROIE, FUNSD and Nomenclatures - characterized in the chapter 3. Another important criterion was the percentage of files for both training and tests. Since the training part is one of the most important, the percentage selected was 90% for the first group and 10% for the second. Given the small size of all datasets, the set of images for the validation phase was not taken into consideration.

In this trial, the accuracy was calculated in such a way that the classifications of the predicted values were compared to the annotations from the dataset. The final value is the ratio of equality between the classifications of the predicted text and the text from the ground truth.

### 5.1.1 SROIE

When using the SROIE dataset, the files were divided into 563 files for training and 63 files for testing. With this division, it was possible to evaluate objectively the

performance of the model. During the training phase, Donut has learnt to recognize and interpret the layout of the documents, which has a predominant sequence - first, there is information regarding the name and the address of the shop, followed by the date when the purchase was made and finally the total price.

The plots in the figure 5.1 show the results observed from the investigation. The first graph - figure 5.1(a) - represents the training loss obtained for different numbers of the training epochs, while the second - figure 5.1(b) - and third ones - figure 5.1(c) illustrate the same relation, but between epoch, test accuracy and average Levenshtein distance, respectively.

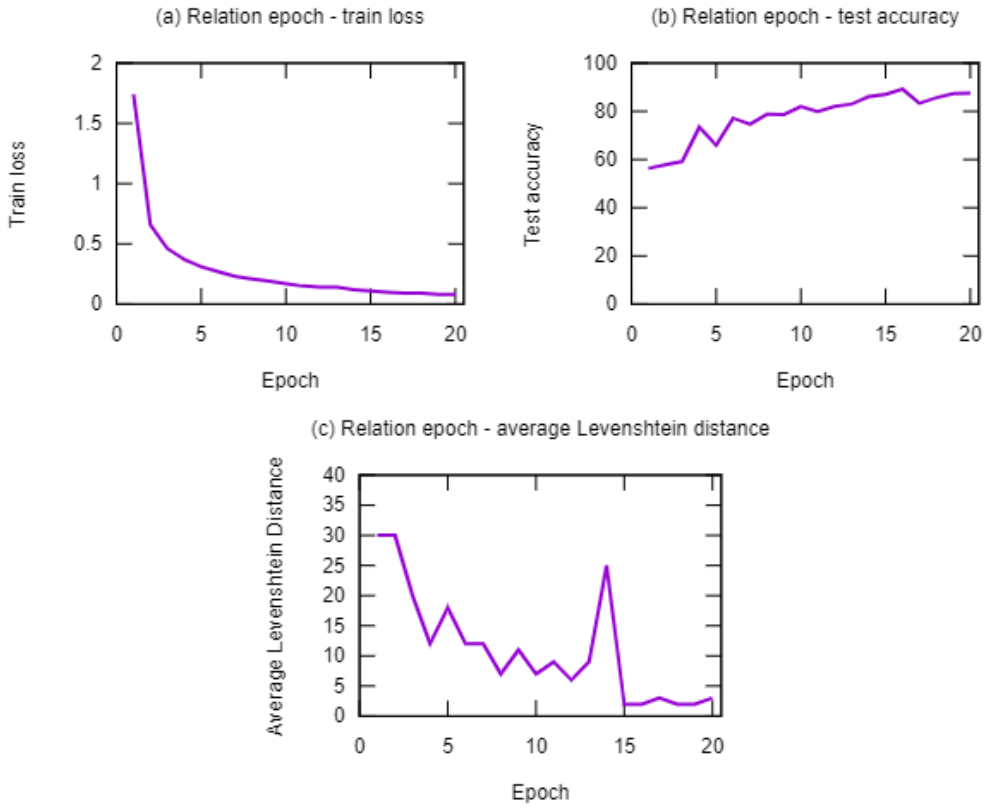


Figure 5.1: Results from Donut experiment with SROIE - 1 to 20 epochs

As depicted in figure 5.1(a) training loss is inversely proportional to the number of epochs. In the beginning, the phenomenon of undertraining makes itself present, where a small number of epochs fails to provide the model with sufficient iterations to learn the patterns in the training data, resulting in larger training losses. Up to epoch 2, the registered values were above one, indicating that the program had some trouble comprehending the document's structure. Over the increase in the specified hyperparameter, the discrepancy between the text in the training data and the predicted one got smaller. The system has had more time to collect the

training files' characteristics and such occurrence had a positive impact, decreasing the differences between the predicted output and the ground truth to results below 0.1. The described events usually lead to the improvement of the test accuracy, which was the scenario observed in this trial.

There were small fluctuations in the trials with epoch values ranging from 1 to 5 and 15 to 20 - figure 5.1(b). In other words, the test precision has presented an irregular behaviour for those ranges of dataset cycles. One possible reason that has caused this response was the value attributed to batch size. As stated at the beginning of the chapter, this variable refers to the number of data rows processed during each training step. From the batch size, it is possible to calculate the absolute frequency at which the internal model parameters are updated since it happens right after each batch group is passed at one time through the network. Because the pool of training images had 593 files, this means that the weights were modified about 297 times. Such number of weight modifications can lead to overfitting, which indicates that the algorithm has learnt all the details of the training data, including the sections that don't represent the true properties of the scanned documents. Apart from that, it can be stated that the variation in the test accuracy had an increasing trend.

The first point to be highlighted is that the model could correctly predict and classify more than 50% of the data having only one epoch. The noticeable increase over time shows that Donut learnt the dataset and thus improved its performance, which is also supported by the behaviour of the average Levenshtein distances in the figure 5.1(c). The metric starts with values reaching 30, which is an understandable result given the quantity of epochs. With the number of training series going up, the final line had a decreasing inclination up to 13 epochs. There was an unexpected increase in the number of text discrepancies in the trial with 14 epochs, which was attributed to random data shuffling inherent in deep learning processes. After that, the metric continued declining, reaching values under 5.

It can be concluded that Donut's achievement with the SROIE dataset was outstanding. The average testing precision and training loss were 81.4% and 0.32, respectively, with the best values being reached for 16 epochs - 89.2% for accuracy, 0.1 for loss and 2 for the average Levenshtein distance. In some cases, Donut demonstrated full efficacy by managing to extract all the necessary information for the classification entities' task, as the output from the listing 5.2 displays.

Listing 5.2: Results with 13 epochs

```
1 C:\Windows\System32\Donut>python main.py --epoch-13
2 Reference: {'total': '174.90', 'date': '17/03/2018',
    'company': 'KEDAI PAPAN YEW CHUAN', 'address': 'LOT 276
    JALAN BANTING 43800 DENGKIL, SELANGOR.'}
3 Prediction: {'total': '174.90', 'date': '17/03/2018',
    'company': 'KEDAI PAPAN YEW CHUAN', 'address': 'LOT 276
```

```
    JALAN BANTING 43800 DENGKIL, SELANGOR.'}
```

### 5.1.2 Nomenclature dataset

Testing a state-of-the-art VDU program with a dataset it has never been tested with can answer some questions regarding the generalization capacity, robustness, performance benchmarking and real-world application. When talking about generalization, this type of testing helps to ensure that the implementation is not only restricted to a specific type of data, but can also handle a broader range of inputs. The robustness is another factor to be examined, because it allows developers to recognise potential weaknesses and improve them. Performance benchmarking also plays a crucial role since nowadays there are numerous options for VDU systems and the performance comparison is a way to determine the optimal approach. Lastly, since these systems are often used in the real world, they are constantly exposed to data with different features than the ones from the training session. Therefore, Donut was tested with the dataset Nomenclature.

585 files were used for the training phase and 65 for the testing one. To analyse the information extraction process, the behaviours of 3 variables - train loss, testing accuracy and average Levenshtein distance - were observed.

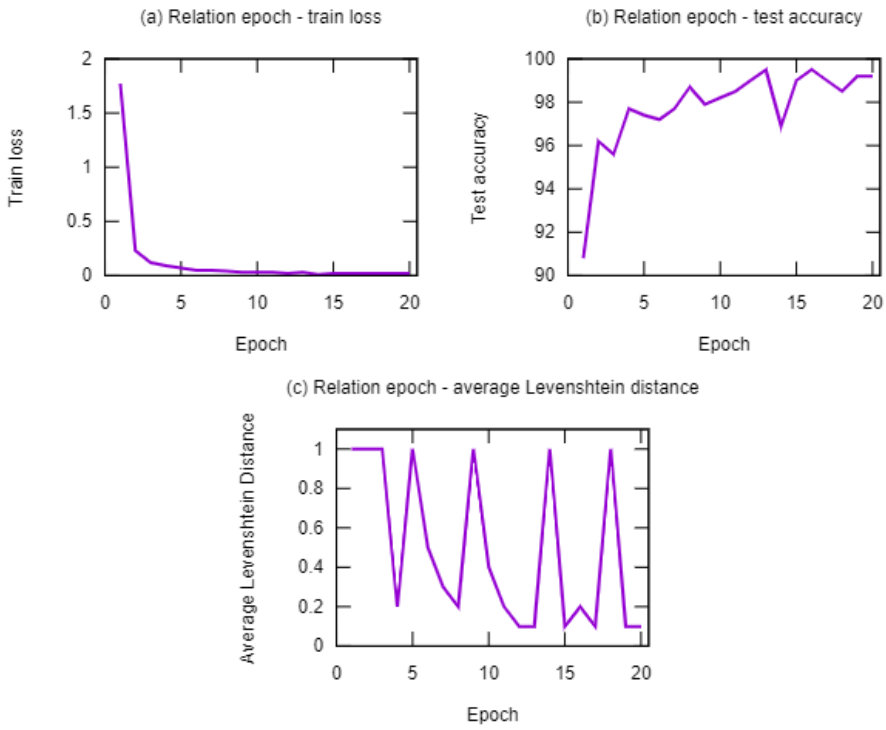


Figure 5.2: Results from Donut experiment with Nomenclature dataset - 1 to 20 epochs

Even though the Levenshtein distance illustrated in figure 5.2(c) does not change uniformly, the registered values were extremely small. This implies that the program remained almost perfectly accurate in its predictions during the experiments with 20 different epochs. When trained with 1, 2, 3, 14 and 18 epochs respectively, there were significant spikes in error followed by a decrease in the next epochs, caused by the occurrence of noisy data. However, given the fact that the dataset contains characters that are not so readable for a computer machine, it is feasible to state that Donut was successful in predicting the ground truth, which is further supported by the fact that the average Levenshtein distance was equal or smaller than 1 through all trials. In summary, the model showed great execution in learning the dataset's layout and consequently minimizing the losses, as the reduction of the training losses reflects - figure 5.2(a). Such facts were also reflected in the test accuracy behaviour.

Concerning the variable above mentioned, the output was greater than 90% in all trials - figure 5.2(b) - meaning that most of the classifications for the predicted nomenclatures were the same as those from the ground truth. In other words, Donut could decipher the characters and provide valid information, as written in the examples in the listing 5.3.

Listing 5.3: Results with epoch equals to 4 and 16

```
C:\Windows\System32\Donut>python main.py --epoch 4
Reference: {'second_sector': 'e', 'roman': 'II',
    'first_sector': 'b', 'colonne': 'OC', 'area': 'Nieder
    Neudorf', 'arabic': ''}
Prediction: {'second_sector': 'e', 'roman': 'XIV',
    'first_sector': 'b', 'colonne': 'OC', 'area': 'Nieder
    Neudorf', 'arabic': '11'}
C:\Windows\System32\Donut>python main.py --epoch-18
Reference: {'second_sector': 'g', 'roman': 'VI',
    'first_sector': 'c', 'colonne': 'OC', 'area': '',
    'arabic': '28'}
Prediction: {'second_sector': 'g', 'roman': 'VI',
    'first_sector': 'c', 'colonne': 'OC', 'area': '',
    'arabic': '28'}
```

The written findings showed significant precision and thus the framework is a very versatile tool for the recognition and classification of a set of data with the particular feature that the text appears to be handwritten rather than machine-generated.

### 5.1.3 FUNSD

The experiment with the FUNSD dataset was done in such a way that from 199 files, 179 were used to train the model and 20 were used for testing purposes. The FUNSD dataset exhibits some differences compared to the two datasets mentioned earlier.

The first one is the amount of variation in the documents. The papers were extracted from different fields, for example, medical reports, customer feedback and event registration. This implies that the structure among them differs. Another particularity is the amount of information written. Usually, a record contains a wide set of details, such as personal, employment, education and demographic information. Since it is important to accordingly classify each entity, the number of their classification will be greater, which will require greater understanding and classification levels from the model. Lastly, another aspect that diverges from the SROIE and Nomenclatures datasets is the format of the annotations. Unlike the first two datasets, each piece of information in the FUNSD images is stored with some additional details, as shown in the listing 1. These three aspects have had a significant influence on the training and testing phases of Donut. The findings are represented below - figure 5.3.

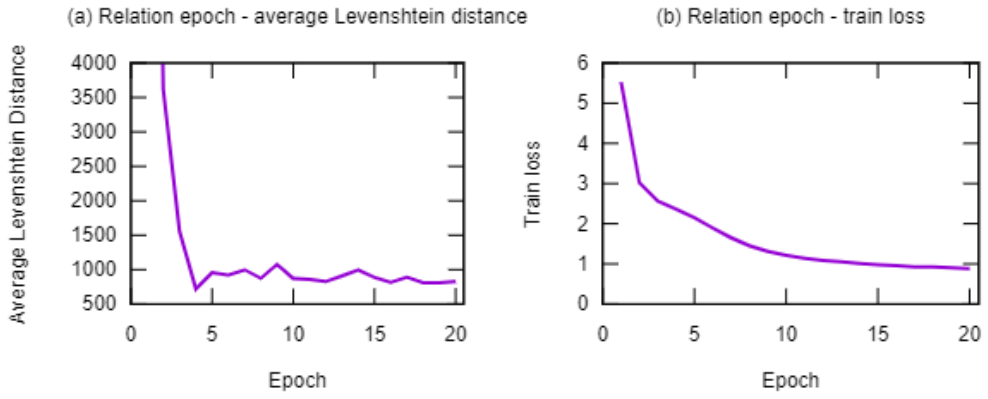


Figure 5.3: Results from Donut experiment with FUNSD dataset - 1 to 20 epochs

As expected, the training loss had a decreasing tendency as the number of epochs increased - figure 5.3(b). However, the values were remarkably high compared to the training losses from SROIE and Nomenclature datasets. Loss during the training process is the quantitative tracking of the model's progress in learning the training dataset. In simpler terms, the training loss represents the discrepancy between the training-predicted content and the ground truth. Up to the trial with 15 epochs the registered values were above 1 indicating that the framework struggled to retain the characteristics of the images. As a result, the test predictions with epochs from 1 to 20 were not accurate, which negatively contributed to the results of the Levenshtein metric.

The plot in figure 5.3(a) represents the average amount of character changes necessary to change the predicted string into the text from the document. The registered values presented a faltering behaviour and consistently surpassed 500, which indicates that there were big differences between the two texts previously

mentioned. Even though the number of iterations got bigger, giving the model more time to analyse and learn the layout of the papers, the complexity of the dataset made the prediction process difficult, resulting in unsatisfactory performance in character recognition. Additionally, Donut was not able to correctly classify the entities - listing 5.4 - and because the program compares the reference information to the predicted one to calculate the final accuracy, the registered values remained at 0% throughout all trials.

Listing 5.4: Results with epoch equals to 8

```
C:\Windows\System32\Donut>python main.py --epoch-8
Prediction: {'words': {'text': 'Date:', 'box': ['103',
    '137', '137', '137']}, 'text': 'Date:'}
C:\Windows\System32\Donut>
```

The previous listing represents the extracted text from a project status report. Within the record, there is information regarding the project's name, the project's leader and the general project description. Although the program was not able to predict all the details written in the paper, it did the recognition of some characters.

The observed outcomes underscore the model's inability to effectively generalize across the dataset when dealing with documents that contain a wide range of information. One aspect that has caused this issue was the variety of structures among the papers from FUNSD. A solution that could improve the generalization would be a selection of input files with one specific construction, so that the model could learn and apply it to the testing files. On the other hand, the continuous increment of the number of epochs could lead to the decrement of training loss and consequently better accuracy.

## 5.2 Doc2Graph

Doc2Graph is a visual document understanding method that makes use of OCR implementations. Unlike Donut, Doc2Graph is more demanding in terms of computation power, as it not only makes the recognition of the characters, but also creates a link between the words and represents them through graphs. Due to its organization particularity, trials were executed with epoch values ranging from 1 to 20 using the FUNSD dataset. To adequately capture the structure and content of the documents, the program had additional arguments to add the positional and textual features to the nodes, exploiting the information extracted from the OCR function. Additionally, capturing the spatial relationships between nodes helps to improve the geometric representation of the final graphs, thereby adding an argument to compute the polar relative coordinates between nodes was necessary.

The model was trained and tested with 149 and 50 files, respectively. The values that were taken into consideration for analysis purposes were the average Levenshtein distance, training loss, testing accuracy and AUC. Accuracy was calculated as the ratio of correctly identified labels to the total number of labels in the dataset. The following figures represent the outcome of the experiments. Figure 5.4(a) exhibits the training loss values for each epoch value and figure 5.4(b) illustrates the relationship between epoch and test accuracy.

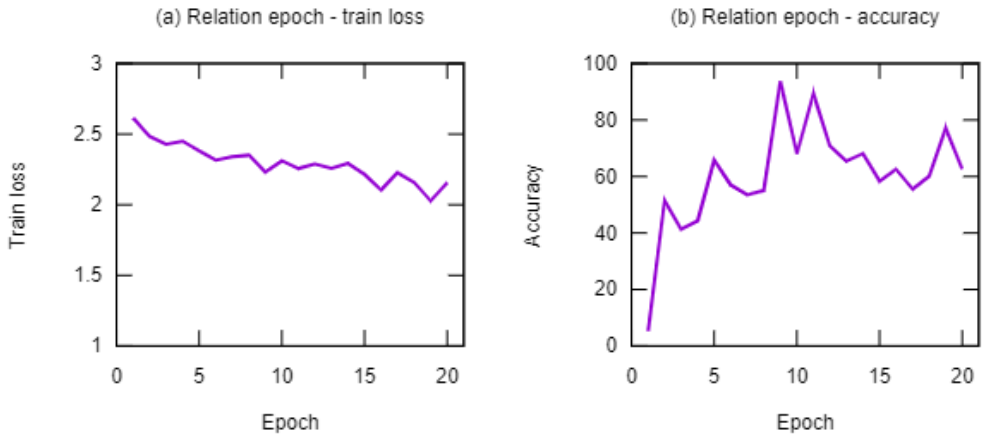


Figure 5.4: Results from Doc2Graph experiment with FUNSD - epoch 1 to 20

Overall, the registered loss had a decreasing tendency throughout the training, going from 2.61 to 2.03. That decrease indicates that the model has continuously learned and improved its performance over more training iterations. In other words, the model became more accurate at predicting the correct outputs as training cycle numbers got higher. This also results in the model converging towards a better solution and achieving a higher level of optimization. That would cause accuracy to have a growing trend.

Yet, looking at the accuracy values across different epochs - figure 5.4(b) - it can be concluded that the accuracy varies irregularly. The first trial registered an accuracy of 5.4%, showing glitches in deeply studying and understanding the structure of the documents. Up to epoch 9, the metric results had an increasing behaviour, reaching a peak of 93.6% accuracy, which suggests that extended analysis time led to improved model comprehension of the documents. Given the complexity of the architecture, the model had some endeavour to precisely generalize the content learned from the training step, causing some epochs to show some improvements and others some difficulties. Another contributing factor that has impacted the differences between the prediction and the ground truth is the variability within the documents of FUNSD. The papers are relevant to a wide range of fields and there-

fore their composition differs. As a result, the model expended additional effort into the correct classification of the entities.

Concerning the AUC score, the records were 50% for all the trials. That informs that Doc2Graph is as good as a system that randomly assigns the entities to their classes, which indicates that the performance of the program was not that satisfactory. Similarly, the Levenshtein distance remained constant across all experiments.

Doc2Graph focuses on the construction of the graph based on the structure of the document and therefore it makes use of the Tesseract OCR engine. Before the definition of the nodes, Tesseract makes the identification of the words, along with some other information, such as their bounding box coordinates. The framework is already pre-trained and since the dataset is the same, the output text remained the same during the experiments. This fact is supported by the consistent average Levenshtein distance of 565 across all twenty tests. Based on the metric result, the recognized texts differ greatly from the ground truth texts, implying potential errors or lack of precision in the recognition process.

Summing up, this framework has shown to be an intermediate option for visual document understanding. The average accuracy was approximately 60.7%, which is neither classified as good nor bad performance. A factor that could make the output better would be a better selection of the dataset. Training and testing the model with forms from one field only would make the knowledge of the structure more accurate and that would reflect on better predictions.

## 5.3 Results comparison

After the collection of the data, the analysis and further comparison with results from other researchers are two important points for observing the VDU methods' flexibility to generalize patterns from the dataset, determining the real-life scenarios where their usage is efficient, and examining the implementation factors that require some improvement. Inasmuch as the importance of the results, this chapter aims to address the mentioned questions regarding the studied models. For that reason, there will be two sections dedicated to the study of the two utilized datasets - SROIE and FUNSD.

The first section will discuss the differences between the output values from Donut when trained and tested with SROIE and CORD, respectively. Although both datasets have a similar layout and are used for the same task, the methods used for the acquisition of the images were different. The images from CORD were collected through crowd-sourcing [19], where a group of people provided photos of receipts they got from restaurants and stores, covering personal details in order to preserve their identity. As for SROIE, the papers were scanned, which provides better visibility conditions of the context, when compared to CORD. Thus, the

analyses of such conditions and the additional language differences will contribute to assessing the model's adaptability. In this situation, some underperformance is expected from the model when fine-tuned with CORD. The final section will focus on the impact of OCR functionalities on recognizing characters. Given the diverse layouts and content structures forms contain, the FUNSD was the dataset chosen for both Donut and Doc2Graph. The results of the evaluation metrics will serve as the base for the comparison of the two implementations.

### 5.3.1 Evaluation of Donut Performance on SROIE and CORD Datasets

Although both datasets were used for information extraction from receipts, there was a small discrepancy in the average accuracy results - 81% for tests with SROIE from the experiments of this paper and 90% with CORD, based on the experiments from the Donut's official article [5].

Regarding the scanned documents, this performance level was influenced by the value of one hyperparameter, which led to overfitting. On the other hand, the accuracy obtained on the CORD dataset was higher. The detailed information on hyperparameters was omitted for fair comparisons, which consequently, makes the process of comparison difficult since the exact reasons behind this discrepancy remain uncertain. However, it is possible that the differences in model fine-tuning could have contributed to this variation. Potentially, Donut was trained with a higher number of epochs, which contributed to a better understanding and, hence to better predictions of the dataset's layout. Further investigation into the specific hyperparameters and preprocessing steps, such as the split of the dataset for training and testing phases, used for the CORD dataset would provide an optimal configuration to use the framework.

By studying these outputs, it can be concluded that Donut fulfilled the task of information extraction. The ratio of the content recognition and classification was above 80%, implying that Donut has the potential to process papers with a small amount of information, even if the document's presentation is not the best.

### 5.3.2 Impact of OCR functionality for text recognition

As the title suggests, this section focuses exclusively on the performances of the text recognition abilities in both Donut and Doc2Graph when used with the FUNSD dataset. The metric included in the evaluation was the Leveshtein distance.

Doc2Graph makes use of a pre-trained OCR model and as such, the Levenshtein distance observed was 565 for all twenty experiments. In contrast, Donut exhibited varying text prediction outputs across different epochs, which indicates that the results were influenced by factors such as the duration of the training phase. Ulti-

mately, the average Levenshtein distance reached 909. The values reveal that even after the training, Donut was not able to predict the content of the images as accurately as the OCR model. Additionally, all Levenshtein distances registered during the experiments for Donut were greater than those for Doc2Graph, which supports the greater reliability of the OCR program, even though it carries higher computational costs. In the majority of real-life applications, the most important variable is the accuracy of the text reader from the files and therefore it can be concluded that the usage of tools like Tesseract is the most recommended option.

Training the OCR-free framework with a different set of hyperparameters could positively impact the outcome text. For instance, increasing the number of epochs to ensure thorough learning and generalization of the dataset's characteristics could lead to better results. However, it would increase the time designed for the training phase. Since there are a set of variables to consider, the decision to choose Donut for text recognition depends on the user's goal.

# Conclusion 6

This research aimed to analyse and compare the performance of the VDU methods for specific types of tasks. For the project's first phase, the methods were theoretically compared for each task described in the first chapter. Among the models, DocFormer is the most prominent one with the best performance for document classification and entity extraction tasks, using RVL-CDIP and CORD datasets, respectively.

Apart from the theoretical analyses, based on their implementations and the representation of the information, two models were trained and tested so that the visual document understanding achievement could be seen and analysed. The first one was Donut, whose versatility was tested using the model with SROIE and Nomenclature datasets, whereas the second set of images had never been used in this program. The results have shown that the framework could correctly extract and categorise a great percentage of the information and it is a tool that with better hyperparameter configurations and bigger datasets, shows even more promising signs of performance. When used with FUNSD, the large amount of detail was a barrier to the model operation. Consequently, the output results were not as solid as the ones obtained from the research with the second chosen model.

Doc2Graph could recognise and classify a higher percentage of entities than Donut. The use of the pre-trained optical character recognition tool - Tesseract - allowed the program to focus only on the training and testing of the entity label detection and labelling, which generated better results. However, the overall evaluation was on an average level. There was still information missing in the predicted text and the area under the receiver operating characteristic has shown that Doc2Graph is as good as a program that distributes the entities to their respective classes in a random manner. Once again, a better selection of the dataset would make the training more consistent and therefore the evaluation would be better.

This research has contributed to a better understanding of some points related to the VDU. The first one is about the dataset, which played a crucial role in the final results. Despite the variation within a dataset helps to prove the ability the model has to face real-world situations, the structure consistency of the files helps to a better understanding of documents and therefore increases the amount of information

recognised and processed. After that, the training phase is another crucial step and since it is influenced by the setting hyperparameters, their values are factors that should be considered before the use of the framework. Lastly, due to the high memory and processor requirements, the features of the device where the framework is run should be acknowledged as well.

# List of abbreviations 7

| Abbreviation | Meaning |
|---|---|
| AP | Average Precision |
| ANLS | Average Normalized Levenshtein Similarity |
| AUC-ROC | Area Under the Receiver Operating Characteristic Curve |
| BROS | Bert Relying On Spatiality |
| CORD | Consolidated Receipt Dataset |
| DLA | Document Layout Analysis |
| DocILE | Document Informat ion Localization and Extraction |
| Donut | Document Understanding Transformer |
| GNN | Graph Neural Networks |
| KIE | Key Information Extraction |
| KILE | Key Information Location Extraction |
| LIR | Line Item Recognition |
| NER | Name Entity Recognition |
| OCR | Optical Character Recognition |
| PIF | Public Inspection Files |
| ROI | Region Of Interest |
| RVL-CDIP | Ryerson Vision Lab Complex Document Information Processing |
| SROIE | Scanned Receipts OCR And Key Information Extraction |
| VDU | Visual Document Understanding |
| VQA | Visual Question Answering |

# Bibliography


1. CAO, Haoyu et al. *Attention Where It Matters: Rethinking Visual Document Understanding with Selective Region Concentration*. 2023. Available from arXiv: 2309.01131 [cs.CV].
2. HSU, Chih-Chung; LEE, Chia-Ming; SUN, Chun-Hung; WU, Kuang-Ming. *OCR is All you need: Importing Multi-Modality into Image-based Defect Detection System*. 2024. Available from arXiv: 2403.11536 [cs.CV].
3. DIAZ, Daniel Hernandez; QIN, Siyang; INGLE, Reeve; FUJII, Yasuhisa; BISSACCO, Alessandro. *Rethinking Text Line Recognition Models*. 2021. Available from arXiv: 2104.07787 [cs.CV].
4. APPALARAJU, Srikar; JASANI, Bhavan; KOTA, Bhargava Urala; XIE, Yusheng; MANMATHA, R. *DocFormer: End-to-End Transformer for Document Understanding*. 2021. Available from arXiv: 2106.11539 [cs.CV].
5. KIM, Geewook et al. *OCR-free Document Understanding Transformer*. 2022. Available from arXiv: 2111.15664 [cs.LG].
6. HARLEY, Adam W.; UFKES, Alex; DERPANIS, Konstantinos G. *Evaluation of Deep Convolutional Nets for Document Image Classification and Retrieval*. 2015. Available from arXiv: 1502.07058 [cs.CV].
7. NAWEI, Chen; BLOSTEIN, Dorothea. Blostein, D.: A survey of document image classification: problem statement, classifier architecture and performance evaluation. International Journal of Document Analysis and Recognition (IJDAR) 10(1), 1-16. *IJDAR*. 2007, vol. 10, pp. 1–16. Available from DOI: 10.1007/s10032-006-0020-2.
8. DING, Yihao et al. V-Doc: Visual Questions Answers With Documents. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2022, pp. 21492–21498.
9. HONG, Teakgyu et al. *BROS: A Pre-trained Language Model Focusing on Text and Layout for Better Key Information Extraction from Documents*. 2022. Available from arXiv: 2108.04539 [cs.CL].

10. ŠIMSA, Štěpán et al. *DocILE Benchmark for Document Information Localization and Extraction*. 2023. Available from arXiv: 2302.05658 [cs.CL].

11. YOOCHAN, Moon; LEE, Jinwon; MUN, Duhwan; LIM, Seungeun. Deep Learning-Based Method to Recognize Line Objects and Flow Arrows from Image-Format Piping and Instrumentation Diagrams for Digitization. *Applied Sciences*. 2021, vol. 11, p. 10054. Available from DOI: 10.3390/app112110054.

12. WANG, Jilin et al. *A Graphical Approach to Document Layout Analysis*. 2023. Available from arXiv: 2308.02051 [cs.LG].

13. BINMAKHASHEN, G. M.; MAHMOUD, S. A. Historical document layout analysis using anisotropic diffusion and geometric features. *International Journal on Digital Libraries*. 2020, vol. 21, pp. 329–342. Available from DOI: 10.1007/s00799-020-00280-w.

14. PAKHALE, Kalyani. *Comprehensive Overview of Named Entity Recognition: Models, Domain-Specific Applications and Challenges*. 2023. Available from arXiv: 2309.14084 [cs.CL].

15. LIU, Yinhan et al. RoBERTa: A Robustly Optimized BERT Pretraining Approach. *ArXiv*. 2019, vol. abs/1907.11692. Available also from: https://api.semanticscholar.org/CorpusID:198953378.

16. GEMELLI, Andrea; BISWAS, Sanket; CIVITELLI, Enrico; LLADÓS, Josep; MARINAI, Simone. Doc2Graph: A Task Agnostic Document Understanding Framework Based on Graph Neural Networks. In: *Computer Vision – ECCV 2022 Workshops*. Springer Nature Switzerland, 2023, pp. 329–344. ISBN 9783031250699. ISSN 1611-3349. Available from DOI: 10.1007/978-3-031-25069-9_22.

17. LACASA, Lucas; LUQUE, Bartolo; BALLESTEROS, Fernando; NUÑO, Juan Carlos. From time series to complex networks: The visibility graph. *Proceedings of the National Academy of Sciences*. 2008, vol. 105, no. 13, pp. 4972–4975. Available from DOI: 10.1073/pnas.0709247105.

18. YANG, Chenxiao; WU, Qitian; WIPF, David; SUN, Ruoyu; YAN, Junchi. *How Graph Neural Networks Learn: Lessons from Training Dynamics*. 2024. Available from arXiv: 2310.05105 [cs.LG].

19. PARK, Seunghyun et al. CORD: A Consolidated Receipt Dataset for Post-OCR Parsing. In: 2019. Available also from: https://api.semanticscholar.org/CorpusID:207900784.

20. JAUME, Guillaume; EKENEL, Hazim Kemal; THIRAN, Jean-Philippe. *FUNSD: A Dataset for Form Understanding in Noisy Scanned Documents*. 2019. Available from arXiv: 1905.13538 [cs.IR].

21. HUANG, Zheng et al. ICDAR2019 Competition on Scanned Receipt OCR and Information Extraction. In: *2019 International Conference on Document Analysis and Recognition (ICDAR)*. IEEE, 2019. Available from DOI: 10.1109/icdar.2019.00244.

22. PEER, David; SCHÖPF, Philemon; NEBENDAHL, Volckmar; RIETZLER, Alexander; STABINGER, Sebastian. *ANLS* – A Universal Document Processing Metric for Generative Large Language Models*. 2024. Available from arXiv: 2402.03848 [cs.CL].

# Appendix 8

The appendix content consists of a DVD, where 8 folders are properly stored.

- Text_prace - all source files used and the resulting PDF file.
- Aplikace_a_knihovny - directory of all source codes and their configurations.

  In both folders, there are the source code, configurations and requirements files. The folder containing the Donut code has two other folders, which are the two pre-trained models with SROIE and Nomenclature datasets, respectively. Given the memory limitation, the files with pre-trained models were divided into 2 parts.
- Vstupni_data - FUNSD and Nomenclature dataset.

  As for the SROIE dataset, instructions can be found here `https://www.philschmid.de/fine-tuning-donut#4-fine-tune-and-evaluate-donut-model`.
- Vysledky - output files of measured results.

  It contains 4 folders, whose names identify the model and the dataset used.

## 8.1 User Manual

### 8.1.1 Donut

The training and testing of Donut can be done by first setting up a conda environment as shown in the listing 8.1.

Listing 8.1: Initialize Donut environment

```
C:\Windows\System32\Donut>conda create -n donut python=3.8.18 &&
 conda activate donut &&
 pip install torch &&
 pip install pillow &&
 pip install accelerate
```

| Argument | Explanation | Required |
|---|---|---|
| –data | Path to the dataset. | Yes |
| –epoch | Number of training epochs. | No |
| –batch-size | Training batch size. | No |
| –learning-rate | Training learning rate. | No |
| –test | Path to pre-trained model to be tested. | No |
| –train | Path from where the model should be taken to be trained. | No |

Table 8.1: arguments for Donut

After that, the user can start the model by accessing the folder *Donut* from the DVD in the appendix and running the file *main.py*, with the arguments described in the table 8.1. An example of command execution is written in the listing 8.2.

Listing 8.2: Example of command to run Donut

```
C:\Windows\System32\Donut>python main.py --data sroie --epoch 10
    --train naver-clova-ix/donut-base --test trained-model
```

A model trained with 20 epochs for the SROIE dataset can be found in the the folder *trained-model-sroie-epoch-20*. The datasets are also included in the mentioned folder, so that it is easier to run the model. More information can be found on the following website.

### 8.1.2 Doc2Graph

In order to test and train Doc2Graph, it is necessary to set up the initial conda environment - listing 8.3 - and install all the packages written in the file *requirements.txt*.

Listing 8.3: Initialize conda environment and install of some packages for Doc2Graph

```
C:\Windows\System32\Donut>conda create -n doc2graph python=3.9
    ipython cudatoolkit=11.3 -c anaconda &&
 conda activate doc2graph
C:\Windows\System32\Donut> pip install torch==1.11.0+cu113
    torchvision==0.12.0+cu113 torchaudio==0.11.0
    --extra-index-url &&
https://download.pytorch.org/whl/cu113 &&
pip install dgl-cu113 dglgo -f
    https://data.dgl.ai/wheels/repo.html &&
pip install setuptools-git-versioning && pip install -e . &&
pip install https://github.com/explosion/spacy-
models/releases/download/en_core_web_lg-3.3.0/en_core_web_lg-
3.3.0.tar.gz
```

The next step consists of creating the project folder structure and downloading the data - listing 8.4.

Listing 8.4: Initialize project environment

```
C:\Windows\System32\Donut>python src/main.py --init
```

Once done, the user should go to the directory *Doc2graph* from the DVD and run the python file *src/main.py* along with some commands from the table 8.2. More information related to the arguments and the model can be found in the Doc2Graph GitHub repository.

| Argument | Explanation |
|---|---|
| –add-embs | Add textual features to graph nodes. |
| –add-eweights | Add polar relative coordinates between nodes to graph edges. |
| –add-geom | Add positional features to graph nodes. |
| –node-granularity | Choose the granularity of nodes to be used. |
| –test | If given, it skips the training step. |
| –weights | When testing, it provides the weight file relative path. |

Table 8.2: Example of arguments to run Doc2Graph

The next listing illustrates the comamnd used to run the training step of the model used in the experiments.

Listing 8.5: Command used to run Doc2Grpah

```
C:\Windows\System32\Donut>python src/main.py --add-embs
    --add-eweights --add-geom
```